REPORT ON THE COLLABORATIVE RESEARCH: ENABLING ON-DEMAND SCIENTIFIC WORKFLOWS ON A FEDERATED CLOUD

CRADA FRA 2014-0002 / KISTI-C14014

STEVEN TIMM, GABRIELE GARZOGLIO,
FERMI NATIONAL ACCELERATOR LABORATORY

SEO-YOUNG NOH, HAENG-JIN JANG,
KISTI

Table of Contents

1. EXECUTIVE SUMMARY

The Fermilab Grid and Cloud Computing Department and the KISTI Global Science experimental Data hub Center are working on a multi-year Collaborative Research And Development Agreement. With the knowledge developed in the first year on how to provision and manage a federation of virtual machines through Cloud management systems, in this second year, they have enabled scientific workflows of stakeholders to run on multiple cloud resources at the scale of 1,000 concurrent machines. The demonstrations have been in the areas of (a) Virtual Infrastructure Automation and Provisioning, (b) Interoperability and Federation of Cloud Resources, and (c) On-demand Services for Scientific Workflows. This is a matching fund project in which Fermilab and KISTI will contribute equal resources.

2. INTRODUCTION

The Cloud computing paradigm has revolutionized the approach to Information Technology in many sectors of society, from telecommunication to the military to science. In particular for science, national laboratories, computing centers and universities in many countries are adapting their current paradigm on distributed computing, complementing the Grid model of federated resources [1,2] with the Cloud model of on-lease dynamically instantiated resources from private and commercial entities.

For institutions such as KISTI and Fermilab, the focus on Cloud computing is dictated by the need to support ever more effectively individual large communities (e.g. the LHC experiments) together with many medium size ones (e.g. the Intensity Frontier experiments and the STAR experiment). With each community requiring slightly different configurations for their computational environment, the use of dynamically instantiated virtual machines managed through a Cloud layer becomes an attractive solution to enable such diversity. In addition, in a time where computing budgets are unable to follow the growth of demand from the stakeholders, research institutions are compelled to improve their flexibility in the allocation of resources. To address the need for flexibility, the characteristic of the Cloud paradigm to instantiate computing services on-demand on a common pool of computing hosts provides a valuable solution.

KISTI and Fermilab have been collaborating on Cloud computing since 2011. In 2013, this collaboration resulted in a formal Collaborative Research And Development Agreement (CRADA) for a multi-year program of work to offer production-quality on-demand computing services to their scientific stakeholders. The vision is to provide layered services on a federation of Clouds, provisioning execution environment on a high-throughput fabric of resources, with the scientists interacting with the Software as a Service layer.

In the first year of collaboration, we focused on proof-of-principle demonstration of basic capabilities: resource provisioning through a few selected Cloud interfaces (Infrastructure as a Service), techniques for virtual machine federation and compatibility, and high-throughput virtualization to build the "fabric" of the computational infrastructure. In this second year, we expanded the work on provisioning and federation, increasing both scale and diversity of solutions, and we started to build on-demand services on the established fabric, introducing the paradigm of Platform as a Service to assist with the execution of scientific workflows. In the years to come we envision to increase both the diversity of Cloud providers and the scale of utilization, improving our ability to federate resources and deploy ensembles of complex services in support scientific computation.

This report focuses on the deliverables for the second year of the program.

3. TECHNICAL DELIVERABLES

The program of work for the second year of the agreement consisted in demonstrations and studies to run scientific workflows on dynamically provisioned resources at the scale of 1,000 concurrent virtual machines. The work was organized in three major areas: (1) Provisioning; (2) Federation and Compatibility; (3) On-Demand Services.

1. **Virtual Infrastructure Automation and Provisioning** focuses on finding fast and reliable mechanisms to access a large amount of resources on private, community, and commercial cloud providers. The area was organized in 5 major activities:
    a. Infrastructural scalability to 1,000 VM: demonstrate a mechanism which can transparently extend grid jobs into FermiCloud and other clouds at scale of 1000 or more simultaneous virtual machines
    b. Scientific workflows scalability to 1,000 VM: run production scientific workflow of at least 1000 simultaneous virtual machines on federated cloud resources via GlideinWMS [4,8] and cloud web

services API's. This activity demonstrates that workflows for the NOvA experiment can take advantage of Cloud resources for their computational peaks at the scale of 1,000 VM.

c. Cost-sensitive provisioning: Continue development of provisioning algorithms that calculate relative cost of commercial cloud and private cloud provisioning, including the potential for spot pricing. Hao Wu, a PhD student from the Illinois Institute of Technology is focusing his research on this topic. The results of his research this year have been accepted at the MTAGS workshop as a paper jointly authored with KISTI [9].

d. Provisioning of a platform of services: Investigate deploying complicated ensembles of virtual machines in support of scientific workflows. This year, the work focused on the dynamic deployment of an ensemble of Squid services on Amazon Web Services. Relying on a cache discovery system (Shoal), the ensemble acts as a discoverable platform for web-based data caching. This infrastructure is used in conjunction with the CERN VM File System (CVMFS) to provide scalable access to software applications on Grid and Cloud platforms.

e. Idle VM detection improvements: we investigated whether it was necessary to add additional functionality to the idle VM detection software developed in the 1st year of the CRADA program, making it publically available to the OpenNebula ecosystem, in case. We are finalizing the investigation on whether the idle VM detection system is sufficient for the current level of utilization of the Cloud resources or further improvements are necessary for next year. Our aim is to generalize the system to work on other Cloud management systems, in addition to OpenNebula, and for commercial clouds.

2. **Interoperability and Federation of Cloud Resources** consists in finding a set of virtual image formats and application programming interfaces that can be used by all members of a virtual organization across a heterogeneous infrastructure. The area was organized in 4 major activities:

   a. Authentication / Authorization: perform comparative investigation of X.509 authentication and authorization used by other cloud software providers and federated cloud taskforces. The investigation was accepted as a paper to the IEEE First International Workshop on Cloud Federation Management [10].

   b. VM image portability: develop automatic virtual machine image format conversion service. This activity resulted in the development of an open source tool and related documentation for porting VM images from an Open Nebula system, such as FermiCloud, to Amazon Web Services. The tool removes configurations specific to the hosting environment, such as disk mount point, converts the resulting image in a format compatible with AWS, and automatically transfers the image in the AWS repository for immediate use.

   c. VM distribution: investigate virtual machine distribution methods and distributed object-based storage. This investigation evaluates scalable global storage technologies shared among hosts in a private Cloud (FermiCloud). Within our scope, a global storage is necessary to speed up the deployment of virtual machines to the hosts and to enable virtual machine live-migration. The current technologies (Red Hat clustering) do not scale well. The investigation focused on the CEPH storage system and gave encouraging results, although deployment and operations are expected to be more stable on the upcoming generation of operating systems (Scientific Linux 7)

   d. Cloud interoperability: expand interoperability studies to cover more commercial and community clouds. This study focused on the documentation and proof-of-principle integration of the Google Compute Engine and Microsoft Azure Cloud with the Fermilab computational environment. The goal of this activity was to expand the range of commercial Cloud providers available to our scientific stakeholders, to avoid vendor lock in and optimize workflow execution.

3. **On-demand Services for Scientific Workflows** aims at finding the most efficient methods for scientific grid and cloud computing middleware to distribute data and execution across the WAN to meet the demand. This area was organized in 4 activities:

   a. Data movement on-demand: evaluate strategies for data storage and movement in support of scientific workflows. For virtual machines running at AWS, we evaluated the cost effectiveness of storing data on local storage (S3): taking advantage of the S3 scalability, jobs could store data immediately at the end of the jobs. We compared this with keeping virtual machines waiting on a data transfer queue to move data back to Fermilab storage. In our model, the latter approach was most cost effective. In addition, this activity supported the provisioning a platform of services, discussed in (1), focusing on the deployment of an ensemble web caching servers (Squid), discoverable through a common name server (Shoal).

b. MPI on Virtual Clusters: continue virtualized MPI computational chemistry workflows with KAIST stakeholders and other interested parties. In the first year of the CRADA we enabled the use of InfiniBand from virtualized nodes, this year we collaborated with Prof. Yousung Jung to try the system for quantum chemistry and quantum mechanical periodic solid computations. The demonstration exposed the need for further optimization of the infrastructure, in order to achieve the expected scalability.
c. User-oriented best practices: develop best practices for running workflows on public clouds with limited Internet accessibility and significant data bandwidth charges. As the number of communities using Cloud resources for production activities is still relatively small, direct interaction with the users has been the preferred way to discuss best practices on the Cloud. We envision writing documentation on this topic in the next year of our collaboration.
d. Fabric improvements: do high-bandwidth storage and network fabric research as necessary to support the above workflows. At the scale of 1,000 virtual machines, we did not uncover the need for further tuning of the fabric to support the computational activities.

The following sections describe in more detail these demonstrations and studies.

3.1. Virtual Infrastructure Automation and Provisioning

Infrastructural and scientific workflows scalability to 1,000 VM

We have previously identified the cosmic-ray Monte Carlo simulation of the NOvA experimental detector as a workflow that is ideally suited for commercial cloud computing due to almost no input files, a modest 1GB output file, and being predominantly CPU bound. We have previously run up to 100 simultaneous virtual machines on AWS and up to 150 jobs on FermiCloud (50 VM's at 3 jobs per VM). On Amazon Web Services the limiting factor had been bandwidth to an auxiliary Squid caching server on the site of Fermilab. It was also cumbersome to create a new working image on AWS due to our requirements for use of special kernel modules. The "provisioning of a platform of services" activity, described below, successfully addressed the caching issue. The virtual machine image format conversion tool, described below in the "virtual machine image portability" section, simplified the process of creating or changing the stock virtual machine. On AWS we typically run machine type m3.large, which has the capacity to run two NOvA jobs in parallel and features a 32GB SSD-based Instance store.

To expand FermiCloud to the capacity of running 1000 Virtual Machines the following steps were necessary: 140 Dell PowerEdge 1950 worker nodes (8 CPU cores each, 16GB of RAM) were made available for the purpose of running batch jobs on the cloud. We used a locally routable private network for the virtual machines to access the Fermilab site networks. Those few files that we needed from off-site were accessed via a Squid proxy server. We also deployed OpenNebula 4.8 with X.509 authentication. This implements a much more reliable and fully-featured emulation of Amazon EC2. We used our Bluearc NFS server as the image repository, copying the image from the NFS server to the disk and starting it there.

Our worker node images on FermiCloud start with the "Golden Image", which is a minimal Linux installation, and add the extra software packages needed to run a job at boot time of the image. This is done via a series of contextualization scripts that use a one-time application of the Puppet configuration system. On the public network this takes less than a minute to complete. On the routable private network, where we are using a web proxy to access off-site files via http, it can up to 10 minutes for a single VM and significantly longer if many are initializing in parallel on the same node. For the second phase of this test, we used a virtual machine with these configurations pre-applied, which significantly helped the launch and fill times.

The native command line utilities of OpenNebula 4.8 can fill the 1000-VM cluster in approximately 30 minutes. In production we use GlideinWMS and HTCondor for VM provisioning and management. HTCondor creates a new ssh key-pair for each virtual machine submitted so that it can uniquely identify them. OpenNebula 4.8 has a hard limit of 300 key-pairs that can be stored per user. An error occurs on each VM submission after that, but it is possible to have HTCondor continue after the error and thus fill the cluster to the full capacity of 1000 virtual machines. It is also necessary to aggressively prune the MySQL database otherwise some queries (DescribeInstances) will time out.

After filling the clouds with 1,000 test jobs, we engaged the NOvA users to submit a real workflow that in total consists of 20000 files to process. We processed these files on AWS and FermiCloud simultaneously, reaching the limit of 1000 simultaneously running VMS (1 job/VM) on FermiCloud and 1000 simultaneously running jobs (2 jobs/VM) on AWS.

The Amazon AWS and FermiCloud private network virtual machines have been added to our production job submission servers as part of this work and we will keep them available for further user access in a service that will be known as the "On-Demand Batch Slot Instantiation Service"

Cost-sensitive provisioning

The design goal of a cost-sensitive algorithm is to enable automatic provision of resources for different scientific applications so that the QoS of the scientific application is met and the operational cost is minimized. The main challenge of designing such an algorithm consists in deciding when and where to allocate resources so that the goals are met. In this study [9], we developed a mechanism to automatically train the VM launching overhead reference model [11]. Based on the virtual machine launching overhead reference model, we have developed an overhead-aware-best-fit (OABF) resource allocation algorithm to help the cloud infrastructure reduce the average VM launching time. This OABF algorithm has been implemented for FermiCloud as a plug in of the vcluster system [5] developed by KISTI. The experimental results indicate that the OABF can significantly reduce the VM launching time (reduced VM launch time by 4 minutes on average) when many VMs are launched simultaneously.

Provisioning of a platform of services

In year 1 we launched individual instances of virtual machines. In this year 2, we started to deploy them as an ensemble with the services that they rely on for scalability. This year we focused on web caching [24], which is our most pressing need in running remotely. We used the Shoal system, developed at the University of Victoria, for discovering Squid servers. It consists of three major components:

1. Shoal Agent: it is on each remotely-deployed Squid server and advertises its location to the central server via the RabbitMQ messaging protocol;
2. Shoal Server: it collects the information on the availability and location of the squids ;
3. Shoal Client: it runs on each worker node virtual machine to query the Shoal Server and then modifies the configuration of the virtual machine appropriately with the current list of squid servers

All components of squid, the shoal agent and client are automatically deployed via puppet apply scripts at boot time, using the cloud-init service of AWS.

As part of this investigation we learned about auto-scaling groups, which are available both in AWS and in Google Cloud, and are currently testing a load balancing system that brings multiple squid servers up or down on demand. For the workflows that we currently run on AWS, a single remote squid is sufficient to handle the load. In year 3 we envision deploying submission nodes on demand and storage transfer servers.

3.2. Interoperability and Federation of Cloud Resources

Virtual machine image portability

The current Cloud Management and Virtualization technologies support a number of diverse Virtual Machine formats, not all compatible. VM image conversion is necessary in many cases to instantiate a VM from a Cloud system to another. For our use cases, this happens commonly when transferring a "golden" image (i.e. a VM with stable configuration) from FermiCloud to AWS, for example to run scientific workflows locally and on the commercial platform using the exact same computational environment.

To address this problem, in year 1 we developed step-by-step documentation to manually convert between commonly-used desktop virtualization formats and server-based virtualization and cloud solution, such as OpenNebula, OpenStack, and Amazon EC2 [3]. This year, we developed a tool to automate the process [25].

The tool takes as input a golden image from the FermiCloud image storage; the image does not need to be instantiated for the conversion process to work. It first resizes the image to optimize the space utilization and cost when the VM is uploaded to the AWS image storage, then it removes certain Fermilab-specific configurations, such as disk mount points and network configurations. At this point, the image is imported to the AWS storage.

Amazon Machine Images (AMI) support two types of virtualization: paravirtual (PV) and hardware virtual machine (HVM). In general, paravirtual VMs can run on host hardware that does not have support for hardware-assisted virtualization, and can be fitted with special network and storage drivers to better take advantage of the underlying hardware. Historically, PV VMs had better performance than HVM VMs in many cases, but because of enhancements in HVM virtualization and the availability of Solid State Device (SSD) disk drives, this tends not to be the case any longer. PV formats also tend to be smaller in size than their HVM counterparts. Irrespectively, after the import both VM formats can be directly instantiated at AWS.

The tool takes about an hour for the conversion process to complete, resulting typically in VM sizes around 3 GB for PV images and 12 GB for HVM. Early operational experience has identified ways to cut this time significantly.

In the next year of the program, we aim at extending the automation of this process to additional commercial cloud providers, such as Google Computational Engine and Microsoft Azure.

Virtual machine distribution

We initially planned to evaluate OpenStack's Swift object store, the GlusterFS system, and the Ceph system. All of these are distributed object storage systems. Given CERN's recent success in deploying Ceph as the back-end image storage for their large OpenStack system [22], we decided to focus on evaluating this system. We found it to be a very stable system and easy to operate and maintain. We successfully used Ceph to run virtual machines on network-attached remote block devices and also to import and export full images to and from local VM host disks [23]. We found the Ceph system to be very stable in operation and fault-tolerant against single machine failures. As Red Hat Enterprise Linux 7/Scientific Linux 7 becomes more widespread we expect that it will become much more straightforward to deploy and maintain in production and we intend to continue planning towards that deployment.

Cloud interoperability

In order to increase the diversity of resources in our Federation, we have continued our investigation of Application Program Interfaces from major Commercial Cloud providers. Our goal is to enable the use of different providers to run our scientific workflows. Different providers might be more effective in terms of performance or cost to address the specific needs of our scientific workflows. This strategy also mitigates the risk of vendor lock-in. In year 1 we successfully tested Amazon EC2 [3]. This summer we tested the Google Compute Engine and the Microsoft Azure Cloud. We produced a manual [26] for beginning users to start new virtual machines on the Google Compute Engine using the web GUI or by using a Python script with the associated Python bindings. We also collected information on how to start virtual machines on the Microsoft Azure Cloud. For bulk usage of either of these clouds, support will have to be added to HTCondor and GlideinWMS. HTCondor at one point supported the Google Compute Engine but within the past year the API has changed sufficiently that the interface will have to be totally redone and we have supplied the info to the developers on how to do it. We have also identified the work that will have to be done with the Microsoft Azure cloud to get HTCondor to support that. In the next year of the program, we plan to evaluate the integration of HTCondor with the OpenStack Native Cloud Interface, which is being integrated this year.

3.3. On-demand Services for Scientific Workflows

Data movement on-demand

We have evaluated the cost of two different strategies to transfer the output of 1,000 jobs running on AWS to a remote archive (e.g. the Fermilab Archival facility). The two strategies incur in different costs depending on parameters such as the effective bandwidth of the transfer, the total size of the output, and the hourly cost of virtual machines. The two strategies are described below:

1. When a job finishes its processing phase and is ready to transfer its output, the job initiates a direct transfer to the external archival facility. We assume that multiple jobs finish approximately at the same time and the archival service allows only a limited number of transfers to avoid overload; the archival service queues up the requests that cannot be immediately served. Jobs that are waiting still incur the regular cost of running a VM at AWS. Eventually, all jobs transfer data back to archive.
2. Finished jobs transfer output to S3, the scalable storage service at AWS. Because of its scalability, all transfers can happen approximately at the same time and all VM can terminate afterwards: there is no cost due to idle jobs waiting to transfer data. Storing data in S3, however, has a cost depending on data size and storage usage time. All data is transferred asynchronously to the archive, then the data can be erased.

In our model, depending on the specific values of the parameters, sometimes the cost of strategy (1) is smaller than (2), sometimes vice versa. For example, with 1,000 VM each transferring a 2 GB output, considering a bandwidth to storage of 10 Gbps, strategy (1) costs $302 and strategy (2) $275. With the same conditions but with an output size of 1 GB per VM, the costs reverse, with strategy (1) costing $155 and strategy (2) $151.

This investigation, did not consider costs reductions on outbound data transfer from AWS through ESNet (the DOE research network provider), which was established at a later time. This study can be further refined and the strategy reevaluated as the costs of AWS change. It seems clear, however, that using S3 can be cost effective, depending on the circumstances and may be worth implementing for year-3 of our collaboration. Making use of S3 caching, in fact, will require changes to our data handling software.

MPI on Virtual Clusters

In year 1, we enabled the use of InfiniBand interconnectivity from Virtual Machines for node-to-node inter-process communication [13,19]. This year we worked with Prof. Yousung Jung from KAIST to evaluate the deployment with out-of-the-box network settings. The evaluation uncovered that further tuning of the communication layer is necessary to use the infrastructure for heavily parallel jobs. As a benchmark, Prof. Jung used two scientific applications based on quantum chemistry and quantum mechanical periodic solid models, using the Message Passing Interface (MPI) for inter-process communication. When run on a local reference cluster, tuned for efficient use of the InfiniBand interface, the running time of the overall computation was reduced almost linearly by growing the number of allocated processors (e.g. for quantum chemistry calculations, 11,693 seconds for 8 cores in 1 node vs. 5,354 seconds for 16 cores from 2 nodes). On FermiCloud, the same result could not be reproduced as the computational time increased by adding cores from different nodes, exposing that the inter-process communication dominated the computational time. Although standard benchmarks (HPL Linpack) on FermiCloud showed that virtualized InfiniBand through SR-IOV drivers had performance equivalent to bare metal especially for large messages, the experience running scientific code demonstrated that further tuning of the communication layer is necessary to efficiently support generic scientific applications. We note that Infiniband work that was done by our student Tiago Pais and reported in the first year of the CRADA has now been included as part of a larger journal article by I. Sadooghi et al [13], which is a comparative study of launch times and HPC performance between AWS and FermiCloud; this has been submitted to the IEEE Transactions on Cloud Computing.

4. QUALITATIVE AND QUANTITATIVE OUTPUT

The collaboration between Fermilab and KISTI achieved the goals in the statement of work for the second year of the CRADA. Broadly, we have achieved two main results:

1. We have developed techniques and methods to increase the scale of resource provisioning to 1,000 VM and the complexity of on-demand services dynamically deployed in support of scientific computation;
2. We have automated and simplified the formatting and distribution of virtual machines across an increasingly broad federation of diverse Cloud providers.

Quantitative considerations:

This year, the collaboration has worked on 5 papers. It has published a paper at the Workshop on Many-Task Computing on Clouds, Grids, and Supercomputers (MTAGS) [9]; one at the 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2014) [11]; another at the Cloud Federation Management workshop [10], which is part of the Utility and Cloud Computing 2014 conference in London in Dec 2014. It has also submitted another two to the journal "IEEE transactions on Cloud computing" [12,13], receiving positive feedback at this stage of the process. In summary, the two-years program has produced eight papers to date, adding these five to the three published in the first year [6, 7, 19].

The work from year 2 was presented at four talks [11,14,15,16] and is scheduled at three more after Oct 2014 [10,17,18]. Some of these talks will result in additional papers in the proceedings. In summary by early 2015, the overall 2 years program will have been presented at eleven talks, adding these seven to the three from year 1 [19,20,21].

We also made publicly available all documentation produced [23,24,25,26], including this report. In addition, all the code is available from the code repository of the FermiCloud project or github.com. The code consists of the automated virtual machine image format conversion tool [27] and the infrastructure to provision a platform of web caching services based on Squid and Shoal [29]. This code consists of interpreted scripts and puppet manifests that can be used directly without a binary distribution. A third development [28] consists of a plug-in and modified code for the vcluster system [5], developed at KISTI. This code instruments the process of virtual machine provisioning and implements the cost-sensitive provisioning algorithm described above [9, 11].

Qualitative considerations:

Build relationships with cloud facilities in US and Pacific Rim: we have acquired allocations on Amazon Web Service, Google Computational Engine, and Microsoft Azure and we have started exploring potential fare reductions applicable to research institutions. In particular, AWS has recently agreed to discount the cost for outgoing network traffic through ESNet (our main research network provider) and we are in the process of enabling this feature for our traffic. We are also discussing potential grant funding from AWS and Microsoft to bootstrap the use of Cloud computing for scientific computation at the next scalability level (beyond 1,000 concurrent VM). At the same time, we continued our collaboration with Rackspace, University of Texas at San Antonio, and University of Wisconsin at

Madison, acting as stakeholders in the joint work to integrate the native OpenStack "NOVA" interface with HTCondor. This work will enable the native integration of OpenStack deployment, including University of Notre Dame, in the Federation.

Identify and begin to fix potential interoperability problems in cloud Application Programming Interfaces: We have identified defects in the use of the Google Computational Engine API from HTCondor and reported the problems to the HTCondor and Google teams. In addition, we've uncovered defects in the OpenNebula V4 implementation for which we've implemented a work around and reported the problem the development teams.

Create a virtual infrastructure able to interoperate with leading commercial and scientific cloud facilities: through the demonstration of running scientific workflows for the NOvA experiment on 1,000 virtual machines deployed on FermiCloud and AWS, we have effectively created a federated virtual facility for our scientific stakeholders. During the course of the past year, three other Fermilab experiments, the Dark Energy Survey, the MicroBooNe experiment, and the mu2e experiment, have also made use of on-site FermiCloud batch resources, easily leveraging the work that was done to enable NOvA.

5. BUDGET ALLOCATION

A. Fermilab-funded effort: TOTAL INDIRECT COSTS Fermi Research Alliance, LLC (FRA) / Fermilab FY2014 provisional indirect cost rate is currently 70.00% (Salaries – SWF), 15.50% (Travel), and 21.28% (Other Material & Services – M&S) of Modified Total Direct Cost, in accordance with Fermilab's contract with the Fermi Research Alliance, LLC (FRA) and the Department of Energy.
The budgeted amount to contribute $100,000 of direct costs (equivalent to $170,000 with indirect cost) was estimated at 7.5 FTE-months. The table below shows effort until the end of September. We estimate for October an effort of at least ¼ FTE-month, thus the budget estimates were met.

PERSON	ADJUSTED EFFORT, FTE-Months (to SEP 30, 2014)
Bernabeu Altayo, Gerard	0.76
Garzoglio, Gabriele	0.45
Kim, Hyun Woo	3.41
Peregonow, Nicholas	0.22
Timm, Steven	2.42
TOTAL	**7.26**

B. Obligated funds provided by KISTI as of Oct 2014 - Indicative report

3 IIT students for the summer (Xu Yang, Sandeep Palur, Hao Wu)	$33540
1 INFN student for the summer (Alessio Balsini) (labor, housing, transportation) *	$6,000
1 consultant for the summer (Kirk Shallcross)	$32,800
Computing cycles at Amazon Web Services *	$1,579
Travel *	$5,233
Miscellaneous (temp. housing, etc)	$1,088
TOTAL DIRECT COST	$80,230
INDIRECT COST (15.50% on Travel; 21.28% on other M&S; 70.00% on SWF)	$16,770
DOE ADMINISTRATIVE FEE (3%)	$3,000
INDICATIVE TOTAL	**$100,000**

***** Not all obligated money has yet been invoiced by the vendors or paid out. Some expenses (such as travel to KISTI) will be reconciled in November. The complete financial report will be available on December.

6. REFERENCES

[1] Pordes, R. et al. (2007). *The Open Science Grid*, J. Phys. Conf. Ser. 78, 012057.doi:10.1088/1742-6596/78/1/012057.

[2] Kranzlmüller, D., J. Marco de Lucas, and P. Öster. "The European Grid Initiative (EGI)." In *Remote Instrumentation and Virtual Laboratories*, pp. 61-66. Springer US, 2010.

[3] Virtual machine interoperability documentation: http://cd-docdb.fnal.gov/cgi-bin/ShowDocument?docid=5208

[4] Sfiligoi, I., Bradley, D. C., Holzman, B., Mhashilkar, P., Padhi, S. and Wurthwein, F. (2009). *The Pilot Way to Grid Resources Using glideinWMS*, 2009 WRI World Congress on Computer Science and Information Engineering, Vol. 2, pp. 428–432. doi:10.1109/CSIE.2009.950.

[5] Seo-Young Noh, Steven C. Timm, Haeng-jin Jang: *vcluster: A Framework for Auto Scalable Virtual Cluster System in Heterogeneous Clouds*, in Cluster Computing (2013).

[6] Hao Wu, Shangping Ren, Gabriele Garzoglio, Steven Timm, Gerard Bernabeu, Hyun Woo Kim, Keith Chadwick, Seo-Young Noh, Haeng-Jin Jang, *Automatic Cloud Bursting Under FermiCloud*, ICPADS CSS workshop, Seoul, 2013, published in IEEE Xplore Digital Library, DOI: 10.1109/ICPADS.2013.121

[7] S. Timm, K. Chadwick, G. Garzoglio, *Grids, Clouds and Virtualization at Fermilab*, accepted in the Proceedings of the Journal of Physics: Conference Series by IOP Publishing, 2013.

[8] P. Mhashilkar, A. Tiradani, B. Holzman, K. Larson, I. Sfiligoi, M. Rynge, *Cloud Bursting with Glideinwms: Means to satisfy ever increasing computing needs for Scientific Workflows*, accepted in the Proceedings of the Journal of Physics: Conference Series by IOP Publishing, 2013

[9] Hao Wu, Shangping Ren, Steven Timm, Gabriele Garzoglio, Seo-Young Noh, *Overhead-Aware-Best-Fit (OABF) Resource Allocation Algorithm for Minimizing VM Launching Overhead*, 7th Workshop on Many-Task Computing on Clouds, Grids, and Supercomputers (MTAGS) 2014, Nov 2014, New Orleans, Louisiana, USA

[10] Hyunwoo Kim, Steve Timm, *X.509 Authentication/Authorization in FermiCloud*, IEEE 1st International Workshop on Cloud Federation Management, Dec 2014, London, UK

[11] Hao Wu, Shangping Ren, Gabriele Garzoglio, Steve Timm, Gerard Bernabeu, Seo-Young Noh, *Modeling the Virtual Machine Launching Overhead under Fermicloud*, 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2014), May, 2014, Chicago, IL, USA

[12] Hao Wu, Shangping Ren, Gabriele Garzoglio, Steve Timm, Gerard Bernabeu, Keith Chadwick, Seo-Young Noh, *A Reference Model for Virtual Machine Launching Overhead*, submitted in 2014 to the IEEE Transactions on Cloud Computing.

[13] Sadooghi, Iman; Hernandez Martin, Jesus; Li, Tonglin; Brandstatter, Kevin; Zhao, Yong; Maheshwari, Ketan; Raicu, Ioan; Pais Pitta de Lacerda Ruivo, Tiago; Garzoglio, Gabriele; Timm, Steven, *Understanding the Performance and Potential of Cloud Computing for Scientific Applications*, submitted in 2014 to IEEE Transactions on Cloud Computing TCC-2013-11-0278.R1

[14] G. Garzoglio, *On-demand Services for the Scientific Program at Fermilab*, International Symposium on Grids and Clouds 2014 (ISGC 2014), March 2014, Taipei, Taiwan

[15] S. Timm, G. Garzoglio, *FermiCloud On-demand Services: Data-Intensive Computing on Public and Private Clouds*, HEPiX Spring 2014 Workshop, May 2014, Annecy-le-Vieux, France

[16] S. Timm, G. Garzoglio, *FermiCloud On-demand Services: Data-Intensive Computing on Public and Private Clouds*, Computing Technique Seminar at CERN, May 2014, Geneva, Switzerland

[17] S. Timm, *Authentication, Authorization, and Federation in OpenNebula with FermiCloud*, OpenNebula Conf 2014, Dec 2014, Berlin, Germany

[18] S. Timm, et al, *Cloud services for the Fermilab scientific stakeholders*, submitted to Computing in High-Energy Physics 2015 (CHEP15), Apr 2015, Okinawa, Japan

[19] Tiago Pais Pitta de Lacerda Ruivo, Gerard Bernabeu, Gabriele Garzoglio, Steve Timm, Hyunwoo Kim, Seo-Young Noh, Ioan Raicu, *Exploring Infiniband Hardware Virtualization in OpenNebula towards Efficient High-Performance Computing,* 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2014), May, 2014, Chicago, IL, USA

[20] Timm, S. (2013, Sep 23) *High Throughput and Resilient Fabric Deployments on FermiCloud*, invited talk at ISC Cloud 2013 symposium, Heidelberg, Germany. https://cd-docdb.fnal.gov:440/cgi-bin/ShowDocument?docid=5202

[21] Timm, S. (2013, Sep. 24) *Enabling Scientific Workflows on FermiCloud using OpenNebula*, keynote talk at OpenNebulaConf 2013, Berlin, Germany. https://cd-docdb.fnal.gov:440/cgi-bin/ShowDocument?docid=5203

[22] Daniel van der Ster, Arne Wiebalck, *Building an organic block storage service at CERN with Ceph*, presented at the 20th International Conference on Computing in High Energy and Nuclear Physics (CHEP2013), pub. Journal of Physics: Conference Series 513 (2014) 042047, doi:10.1088/1742-6596/513/4/042047

[23] Fermilab DocDB: Xu Yang - Ceph Documentation – http://cd-docdb.fnal.gov/cgi-bin/ShowDocument?docid=5428

[24] Fermilab DocDB: Sandeep Palur - Squid and Shoal Server Documentation – http://cd-docdb.fnal.gov/cgi-bin/ShowDocument?docid=5428

[25] Fermilab DocDB: Automated Image Format Conversion from FermiCloud to AWS - Documentation – http://cd-docdb.fnal.gov/cgi-bin/ShowDocument?docid=5428

[26] Fermilab DocDB: Alessio Balsini work on Google and Microsoft Cloud – http://cd-docdb.fnal.gov/cgi-bin/ShowDocument?docid=5428
[27] Fermilab DocDB: Automated Image Format Conversion from FermiCloud to AWS - Code – http://cd-docdb.fnal.gov/cgi-bin/ShowDocument?docid=5428
[28] Code repository at https://github.com/philip-wu5/project/tree/fermi ; Printout at the Fermilab DocDB: Hao Wu - Cost-sensitive Provisioning Algorithm - Code – http://cd-docdb.fnal.gov/cgi-bin/ShowDocument?docid=5428
[29] Fermilab DocDB: Sandeep Palur - Coordinated Workflows with Squid - Code – http://cd-docdb.fnal.gov/cgi-bin/ShowDocument?docid=5428